REMARKS

Claims 1-25 remain pending in this Application. By this Amendment, Applicants amend claim 22 to correct a minor informality; amend the specification to correct misstatements of fact, to correct minor informalities and to supply missing reference numerals so as to conform the specification with the drawings as originally filed; and amend Figure 2F of the drawings in order to avoid duplicative use of reference numerals.

In the Requirement for Information (RFI) dated October 7, 2004, the Examiner: (1) requests documentary support for Applicants' contention that [incr Tk] preallocates memory space for option values (*see RFI*, p. 3, ll. 11-13); (2) requests an explanation of the storage of "strings or arrays as options associated with an instance object" using the [incr Tk] language (*see RFI*, p. 3, ll. 10-11); (3) asks whether the statement, "An alternative data structure which has, for example, been supported in the [incr Tk] language allows values to be stored in strings or arrays as options associated with an instance object," on page 1, lines 14-16, of the original specification should be deleted (*see RFI*, p. 3, ll. 3-6); and (4) asks whether the "alternative data structure" discussed in page 1, lines 14-16, of the instant specification preallocates memory space (*see RFI*, p. 2, ll. 24-26). Applicants respond to each of these inquiries in turn below.

The Period for Response

Initially, Applicants note that the Examiner set a shortened statutory period for response to the RFI of one month. This is contrary to M.P.E.P. § 704.13 (8th Ed., Rev. 2, May 2004), which states that "[r]equirements for information under 37 CFR 1.105

made without an action on the merits should set a shortened statutory period of two months for reply." However, Applicants representative telephoned the Examiner on November 11, 2004, to discuss the period for reply. In that conversation, the Examiner indicated that the USPTO's electronic docketing system indicated that the correct shortened statutory period of two months was set for reply to the RFI. Accordingly, this response is believed to be timely filed. Nevertheless, an authorization to charge our deposit account for any necessary extension of the period for reply is given below.

(1) The Examiner requests documentary support for Applicants' contention that [incr Tk] preallocates memory space for option values (*see RFI*, p. 3, ll. 11-13).

In response, Applicants refer to the following documents:

(1) Tcl Developer Xchange, *Library Procedures – Tk_ConfigureWidget manual page, at* http://www.tcl.tk/man/tcl8.2.3/TkLib/ConfigWidg.htm (downloaded November 22, 2004), 7 pages [hereinafter *Tk_ConfigureWidget manual*].

(2) sourceforge.net, *Archetype Base Class for [incr Tk], at* http://incrtcl.sourceforge.net/itk/Archetype.html (downloaded November 22, 2004), 4 pages [hereinafter *Archetype Base Class for [incr Tk]*].

(3) McLennan, Michael J., *Object-Oriented Programming with [incr Tcl] Building Mega-Widgets with [incr Tk], at* http://www.ing.iac.es/~docs/external/tcl/itcl/tutorials/itclitk-a4.pdf, pp. 1-118 (1996) [hereinafter *McLennan*].

(4) Ulferts, Mark L., *[incr Widgets] An Object Oriented Mega-Widget Set*, presented at the Third Annual Tcl/Tk Workshop, Toronto, Ontario, Canada (Jul., 1995) [hereinafter *Ulferts*].

Applicants note that *McLennan* and *Ulferts* were cited by the Examiner with the Office Action dated January 22, 2004. *Tk_ConfigureWidget manual* and *Archetype Base Class for [incr Tk]* are cited on the Form PTO/SB/08 attached hereto. Because these references are cited in response to the Examiner's Requirement for Information, no fee is due therewith. See M.P.E.P. § 704.14(d) (8th Ed., Rev. 2, May 2004). The relevance of each of these documents to the Examiner's inquiries is explained below.

(2) The Examiner requests an explanation of the of the [incr Tk] languages storage of "strings or arrays as options associated with an instance object" (*see RFI*, p. 3, ll. 10-11).

In response, Applicants refer the Examiner generally to *McLennan*, *Ulferts* and *Tk_ConfigureWidget manual*. In particular, *McLennan* discusses the use of options on at least pages 71-99 and 108-113, and *Tk_ConfigureWidget manual* discusses options at least in the section entitled "DESCRIPTION" on pages 2-3 of the printed document.

(3) The Examiner asks whether the statement, "An alternative data structure which has, for example, been supported in the [incr Tk] language allows values to be stored in strings or arrays as options associated with an instance object," on page 1, lines 14-16 of the original specification should be deleted (*see RFI*, p. 3, ll. 3-6).

In response, Applicants refer the Examiner to *McLennan*, pp. 71-99, which discusses the storage of values in strings and arrays as options associated with an instance object. Further, Applicants hereby amend the cited statement to read, "Such a data structure has, for example, been supported in the [incr Tk] language, which allows values to be stored in strings or arrays as options associated with an instance object." By this amendment, Applicants clarify that the [incr Tk] language is an example of a language that preallocates memory space, as evidenced by the references cited herein.

(4) The Examiner asks whether the "alternative data structure" discussed in page 1, lines 14-16, of the instant specification preallocates memory space (*see RFI*, p. 2, ll. 24-26).

In response, Applicants refer the Examiner to *Tk_ConfigureWidget manual* and *Archetype Base Class for [incr Tk].* Specifically, on page 6 of the printed document, *Tk_ConfigureWidget manual* describes the Tk_Offset macro:

> The Tk_Offset macro is provided as a safe way of generating the offset values for entries in Tk_ConfigSpec structures. It takes two arguments: the name of a type of record, and the name of a field in that record. It returns the byte offset of the named field in records of the given type.

Applicants submit that this statement suggests that every option value that can be set on a Tk widget has a unique, preallocated location in the widget's "record" where that option value will be stored. If space were allocated only for options that are actually set, it would not be possible to specify a fixed offset in the record for storing a value based only on knowledge of the type of record and the name of a field. It would also be necessary to take into account the other options (if any) that had already been set on the specific record in question.

Further, on page 2 of the printed document, *Archetype Base Class for [incr Tk]* describes the "itk_component add" method:

> itk_component add *name createCmds ?optionCmds?*
>
> > Creates a component widget by executing the *createCmds* argument and registers the new component with the symbolic name *name*. The *createCmds* code can contain any number of commands, but it must return the window path name for the new component widget.
> >
> > The *optionCmds* script contains commands that describe how the configuration options for the new component should be integrated into the composite

list for the mega-widget. It can contain any of the following commands:

***

keep *option ?option option ...?*

Integrates one or more configuration *options* into the composite list, keeping the name the same. Whenever the mega-widget option is configured, the new value is also applied to the current component. Options like "-background" and "-cursor" are commonly found on the keep list.

***

If the *optionCmds* script is not specified, the usual option-handling commands associated with the class of the component widget are used by default.

Thus, the “itk_component add” command is used to add a component widget to a mega-widget. In calling this method, the “keep” command must be used to specify option names of the component widget that should be connected to option names of the mega-widget. Each option name to be “kept” is specified individually, and, subsequently, if one of these options is set on the mega-widget, the [incr Tk] system will set the same option value on the component widget.

Since each mega-widget instance will have its own collection of component widget instances, it follows that each mega-widget instance builds its own table of “kept” options, whose size is proportional to the number of options that could be set on the mega-widget, regardless of how many of those options are actually set on the mega-widget.

Further, on page 3 of the printed document, *Archetype Base Class for [incr Tk]* describes the "itk_option define" command:

> itk_option define *switchName resourceName resourceClass init ?config?*
>
> > This command is used at the level of the class definition to define a synthetic mega-widget option. Within the configure and cget methods, this option is referenced by switchName, which must start with a "-" sign. It can also be modified by setting values for *resourceName* and *resourceClass* in the X11 resource database. The *init* value string is used as a last resort to initialize the option if no other value can be used from an existing option, or queried from the X11 resource database. If any *config* code is specified, it is executed whenever the option is modified via the configure method. The *config* code can also be specified outside of the class definition via the configbody command.

Thus, the "itk_option define" command provides a mechanism for defining option names on mega-widgets that may not correspond to option names on any of the component widgets. Note that the value of the "init" argument may be "used as a last resort to initialize the option if no other value can be used from an existing option, or queried from the X11 resource database." That the option value can be "initialized" using the "init" argument discloses that a memory location to hold the option value is preallocated. Because this location exists, it must contain some value, so the init value may be used as a last resort for this purpose. If a preallocated location did not already exist, it could not be "initialized."

Thus, Applicants submit that the "alternative data structure" supported by [incr Tk] does preallocate memory space for option values. Consequently, the skilled person would understand that the statement on page 1, lines 16-19, of the original specification was in error, and Applicants respectfully request that the rejections of claims 1-25 under 35 U.S.C. § 103(a) based on this erroneous statement be withdrawn and the claims allowed.

CONCLUSION

In view of the foregoing amendments and remarks, Applicant respectfully requests reconsideration and reexamination of this application and the timely allowance of the pending claims.

Please grant any extensions of time required to enter this response and charge any additional required fees to our deposit account 06-0916.

Respectfully submitted,

FINNEGAN, HENDERSON, FARABOW,
GARRETT & DUNNER, L.L.P.

Dated: December 6, 2004

By: [illegible]
John M. Mulcahy
Reg. No. 55,940

Attachments: Replacement Drawing Sheet 5/25 (one sheet) showing Figure 2F;
Form PTO/SB/08..

AMENDMENTS TO THE DRAWINGS:

The attached Replacement Drawing Sheet (Sheet 5/25) includes changes to Figure 2F. Specifically, reference numerals “43” and “45” are changed to –44– and –63–, respectively, in order to avoid duplicative use of reference numerals.

Attachments: Replacement Drawing Sheet 5/25 (one sheet) showing Figure 2F.